Exploiting special structure in semidefinite programs: an overview

Etienne de Klerk

Tilburg University, The Netherlands

MOSEK Workshop, October 6, 2014
Primal problem

\[
\begin{align*}
\min_{X \succeq 0} \text{trace}(A_0 X) \quad &\text{subject to} \quad \text{trace}(A_k X) = b_k \ (k = 1, \ldots, m), \\
\end{align*}
\]

where the data matrices \( A_i \in \mathbb{S}^{n \times n} \ (i = 0, \ldots, m) \) are linearly independent.

- \( \mathbb{S}^{n \times n} \): symmetric \( n \times n \) matrices;
- \( X \succeq 0 \): \( X \) symmetric positive semi-definite.
Primal problem

\[
\begin{align*}
\min_{X \succeq 0} & \quad \text{trace}(A_0 X) \\
\text{subject to} & \quad \text{trace}(A_k X) = b_k \quad (k = 1, \ldots, m),
\end{align*}
\]

where the data matrices \( A_i \in \mathbb{S}^{n \times n} \) \( (i = 0, \ldots, m) \) are linearly independent.

- \( \mathbb{S}^{n \times n} \): symmetric \( n \times n \) matrices;
- \( X \succeq 0 \): \( X \) symmetric positive semi-definite.

Sometimes we will add the additional constraint \( X \succeq 0 \) (componentwise nonnegative).
Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:
Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

- low rank of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)
Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

1. low rank of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)

Further reading:

Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

1. **low rank** of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)

Further reading:


2. aggregate **chordal sparsity pattern** of the $A_i$’s; (Wolkowicz et al, Laurent, ...)
Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

1. low rank of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)

Further reading:


2. aggregate *chordal sparsity pattern* of the $A_i$’s; (Wolkowicz et al, Laurent, ...) Talk by Joachim Dahl later today.
Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

1. **low rank** of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)

Further reading:


2. aggregate **chordal sparsity pattern** of the $A_i$’s; (Wolkowicz et al, Laurent, …) Talk by Joachim Dahl later today.

Further reading:

Structured SDP instances

Three types of structure in the SDP data matrices $A_0, \ldots, A_m$ may be effectively exploited by interior point methods:

1. low rank of $A_1, \ldots, A_m$; (Benson-Ye-Zhang, DSDP software)

Further reading:

2. aggregate *chordal sparsity pattern* of the $A_i$’s; (Wolkowicz et al, Laurent, …) Talk by Joachim Dahl later today.

Further reading:

3. if the $A_i$’s lie in a low dimensional matrix algebra (this talk).
A set $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$ (resp. $\mathbb{R}^{n \times n}$) is called a matrix *-algebra over $\mathbb{C}$ (resp. $\mathbb{R}$) if, for all $X, Y \in \mathcal{A}$:
Matrix algebras

Definition

A set $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$ (resp. $\mathbb{R}^{n \times n}$) is called a matrix $\ast$-algebra over $\mathbb{C}$ (resp. $\mathbb{R}$) if, for all $X, Y \in \mathcal{A}$:

- $\alpha X + \beta Y \in \mathcal{A}$ for all $\alpha, \beta \in \mathbb{C}$ (resp. $\mathbb{R}$);
Matrix algebras

Definition

A set $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$ (resp. $\mathbb{R}^{n \times n}$) is called a matrix $\ast$-algebra over $\mathbb{C}$ (resp. $\mathbb{R}$) if, for all $X, Y \in \mathcal{A}$:

- $\alpha X + \beta Y \in \mathcal{A}$ $\forall \alpha, \beta \in \mathbb{C}$ (resp. $\mathbb{R}$);
- $X^* \in \mathcal{A}$;
A set $\mathcal{A} \subseteq \mathbb{C}^{n\times n}$ (resp. $\mathbb{R}^{n\times n}$) is called a matrix $*$-algebra over $\mathbb{C}$ (resp. $\mathbb{R}$) if, for all $X, Y \in \mathcal{A}$:

- $\alpha X + \beta Y \in \mathcal{A}$ $\forall \alpha, \beta \in \mathbb{C}$ (resp. $\mathbb{R}$);
- $X^* \in \mathcal{A}$;
- $XY \in \mathcal{A}$. 

Etienne de Klerk (Tilburg University)
Special structure in SDP
MOSEK workshop 4 / 21
**Definition**

A set $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$ (resp. $\mathbb{R}^{n \times n}$) is called a *matrix *-algebra* over $\mathbb{C}$ (resp. $\mathbb{R}$) if, for all $X, Y \in \mathcal{A}$:

- $\alpha X + \beta Y \in \mathcal{A}$, $\forall \alpha, \beta \in \mathbb{C}$ (resp. $\mathbb{R}$);
- $X^* \in \mathcal{A}$;
- $XY \in \mathcal{A}$.

**Assumption**

There is a ‘low dimensional’ matrix *-*algebra $\mathcal{A}_{SDP} \supseteq \{A_0, \ldots, A_m\}$.
Example

The circulant matrices form a commutative matrix *-algebra.

Form of a circulant matrix $C$

$$
C = \begin{bmatrix}
c_0 & c_1 & c_2 & \cdots & c_{n-1} \\
c_{n-1} & c_0 & c_1 & \cdots & \ \\
c_{n-2} & c_{n-1} & c_0 & c_1 & \ddots & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
c_1 & \cdots & c_{n-1} & c_0
\end{bmatrix}
$$

Each row is a cyclic shift of the row above it, i.e. $C_{ij} = c_{i-j} \mod n$ ($i,j = 0, \ldots, n-1$).
Example

The **circulant matrices** form a commutative matrix $\ast$-algebra.

**Form of a circulant matrix $C$**

$$
C = 
\begin{bmatrix}
  c_0 & c_1 & c_2 & \cdots & c_{n-1} \\
  c_{n-1} & c_0 & c_1 & \cdots & \\
  c_{n-2} & c_{n-1} & c_0 & c_1 & \cdots \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  c_1 & \cdots & c_{n-1} & c_0 \\
\end{bmatrix}
$$

Each row is a cyclic shift of the row above it, i.e: $C_{ij} = c_{i-j \mod n}$ ($i,j = 0, \ldots, n-1$).

**Further reading:**

Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra,
Theorem (Von Neumann)

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_{\mathcal{A}}$. 
Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $*$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_\mathcal{A}$. Then $P_\mathcal{A}(X) \succeq 0$. 
Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_{\mathcal{A}}$. Then $P_{\mathcal{A}}(X) \succeq 0$.

This has the following implication for SDP.
Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_\mathcal{A}$. Then $P_\mathcal{A}(X) \succeq 0$.

This has the following implication for SDP.

**Theorem**

*If the primal SDP has a solution, it has a solution in $\mathcal{A}_{SDP}$.***
Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_\mathcal{A}$. Then $P_\mathcal{A}(X) \succeq 0$.

This has the following implication for SDP.

**Theorem**

*If the primal SDP has a solution, it has a solution in $\mathcal{A}_{SDP}$.***

**Proof:** If $X$ is an optimal solution, then so is $P_{\mathcal{A}_{SDP}}(X)$.  

---
Projection onto $\mathcal{A}$

**Theorem (Von Neumann)**

Let $X \succeq 0$ and $\mathcal{A}$ a matrix $\ast$-algebra, and denote the orthogonal projection operator onto $\mathcal{A}$ by $P_\mathcal{A}$. Then $P_\mathcal{A}(X) \succeq 0$.

This has the following implication for SDP.

**Theorem**

*If the primal SDP has a solution, it has a solution in $\mathcal{A}_{SDP}$.***

**Proof:** If $X$ is an optimal solution, then so is $P_{\mathcal{A}_{SDP}}(X)$.

**Consequence**

We may restrict the primal problem to:

$$
\min_{X \succeq 0} \left\{ \text{trace}(A_0 X) : \text{trace}(A_k X) = b_k \quad (k = 1, \ldots, m), \; X \in \mathcal{A}_{SDP} \right\}.
$$
Theorem (Wedderburn (1907))

Assume $\mathcal{A}$ is a matrix $*$-algebra over $\mathbb{C}$ that contains $I$.
Theorem (Wedderburn (1907))

Assume $\mathcal{A}$ is a matrix $\ast$-algebra over $\mathbb{C}$ that contains $I$. Then there is a unitary $Q$ ($Q^*Q = I$) and some integer $s$ such that

$$Q^*\mathcal{A}Q = \begin{pmatrix}
    \mathcal{A}_1 & 0 & \cdots & 0 \\
    0 & \mathcal{A}_2 & & \\
    \vdots & \ddots & \ddots & 0 \\
    0 & \cdots & 0 & \mathcal{A}_s
\end{pmatrix},$$

where each $\mathcal{A}_i \sim \mathbb{C}^{n_i \times n_i}$ for some integers $n_i$. 
Decomposition of matrix *-algebras

Canonical decomposition of a matrix *-algebra $\mathcal{A}$

**Theorem (Wedderburn (1907))**

Assume $\mathcal{A}$ is a matrix *-algebra over $\mathbb{C}$ that contains $I$. Then there is a unitary $Q$ ($Q^* Q = I$) and some integer $s$ such that

$$Q^* \mathcal{A} Q = \begin{pmatrix}
A_1 & 0 & \cdots & 0 \\
0 & A_2 & & \\
& \ddots & \ddots & 0 \\
0 & \cdots & 0 & A_s
\end{pmatrix},$$

where each $\mathcal{A}_i \sim \mathbb{C}^{n_i \times n_i}$ for some integers $n_i$, and takes the form

$$\mathcal{A}_i = \left\{ \begin{pmatrix}
A & 0 & \cdots & 0 \\
0 & A & \cdots & \\
& \ddots & \ddots & 0 \\
0 & \cdots & 0 & A
\end{pmatrix} \mid A \in \mathbb{C}^{n_i \times n_i} \right\} \quad (i = 1, \ldots, s).$$
Illustration: \( \mathcal{A} \subset \mathbb{C}^{2438 \times 2438} \)

Sparsity pattern of a generic matrix in \( \mathcal{A} \subset \mathbb{C}^{2438 \times 2438} \):
Illustration: $\mathcal{A} \subset \mathbb{C}^{2438 \times 2438}$ (ctd.)

Simple components of $\mathcal{A}$ after unitary transformation:
Illustration: $\mathcal{A} \subset \mathbb{C}^{2438 \times 2438}$ (ctd.)

Irreducible components of $\mathcal{A}$ after second unitary transformation:
Example

The circulant matrices:

\[
\begin{bmatrix}
    c_0 & c_1 & c_2 & \cdots & c_{n-1} \\
    c_{n-1} & c_0 & c_1 & \cdots & \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    c_1 & \cdots & c_{n-1} & c_0 \\
\end{bmatrix}
\]

are diagonalized by the unitary (discrete Fourier transform) matrix:

\[
Q_{ij} := \frac{1}{\sqrt{n}} e^{-2\pi \sqrt{-1} ij / n} \quad (i, j = 0, \ldots, n - 1).
\]
Remarks

If a basis of $\mathcal{A}$ is known, the unitary matrix $Q$ may be computed using only linear algebra.
Remarks

If a basis of $\mathcal{A}$ is known, the unitary matrix $Q$ may be computed using only linear algebra.

**Randomized algorithms:**


If a basis of $\mathcal{A}$ is known, the unitary matrix $Q$ may be computed using only linear algebra.

**Randomized algorithms:**


We give a simple prototype algorithm to illustrate the basic procedure ...
Simple block diagonalization algorithm

**Algorithm**

**Input:** A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra $A$. 
Simple block diagonalization algorithm

**Algorithm**

**Input:** A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra $\mathcal{A}$.

1. Compute a (random) $A \in \mathcal{A}$ such that

$$AB_i = B_i A \quad \forall \ i.$$
Algorithm

**Input:** A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra $\mathcal{A}$.

1. Compute a (random) $A \in \mathcal{A}$ such that

$$AB_i = B_i A \quad \forall i.$$

2. Do the spectral decomposition: $A = Q \Lambda Q^*$ where $Q$ is unitary.

**Output:** Block diagonal matrices $Q^* B_i Q$ ($i = 1, \ldots, d$).
Algorithm

**Input:** A basis $B_1, \ldots, B_d$ of a matrix $*$-algebra $\mathcal{A}$.

1. Compute a (random) $A \in \mathcal{A}$ such that

   $$AB_i = B_iA \quad \forall \ i.$$

2. Do the spectral decomposition: $A = Q \Lambda Q^*$ where $Q$ is unitary.

**Output:** Block diagonal matrices $Q^*B_iQ$ ($i = 1, \ldots, d$).

The block sizes equal the multiplicities of the eigenvalues of $A$. 
Assume we have a basis $B_1, \ldots, B_d$ of $\mathcal{A}_{SDP}$.

\[
\min_{X \succeq 0} \{ \text{trace}(A_0 X) : \text{trace}(A_k X) = b_k \quad (k = 1, \ldots, m), X \in \mathcal{A}_{SDP} \}\]
Assume we have a basis $B_1, \ldots, B_d$ of $A_{SDP}$.

$$\begin{align*}
\min_{X \succeq 0} \{ & \text{trace}(A_0X) : \text{trace}(A_kX) = b_k \ (k = 1, \ldots, m), X \in A_{SDP} \}\end{align*}$$

Setting $X = \sum_{i=1}^d x_i B_i$, this becomes:

$$\begin{align*}
\min_{x \in \mathbb{R}^d} \left\{ & \sum_{i=1}^d x_i \text{trace}(A_0 B_i) : \sum_{i=1}^d x_i \text{trace}(A_k B_i) = b_k \ (k = 1, \ldots, m), \sum_{i=1}^d x_i B_i \succeq 0 \right\}.
\end{align*}$$
**SDP reformulation**

Assume we have a basis $B_1, \ldots, B_d$ of $\mathcal{A}_{SDP}$.

\[
\min_{X \succeq 0} \{ \text{trace}(A_0 X) : \text{trace}(A_k X) = b_k \quad (k = 1, \ldots, m), \ X \in \mathcal{A}_{SDP} \} 
\]

Setting $X = \sum_{i=1}^d x_i B_i$, this becomes:

\[
\min_{x \in \mathbb{R}^d} \left\{ \sum_{i=1}^d x_i \text{trace}(A_0 B_i) : \sum_{i=1}^d x_i \text{trace}(A_k B_i) = b_k \quad (k = 1, \ldots, m), \ \sum_{i=1}^d x_i B_i \succeq 0 \right\}. 
\]

- Replace the LMI by $\sum_{i=1}^d x_i Q^* B_i Q \succeq 0$ to get block-diagonal structure.
Assume we have a basis $B_1, \ldots, B_d$ of $\mathcal{A}_{SDP}$.

$$\min_{X \succeq 0} \{ \text{trace}(A_0X) : \text{trace}(A_kX) = b_k \ (k = 1, \ldots, m), X \in \mathcal{A}_{SDP} \}$$

Setting $X = \sum_{i=1}^d x_i B_i$, this becomes:

$$\min_{x \in \mathbb{R}^d} \left\{ \sum_{i=1}^d x_i \text{trace}(A_0 B_i) : \sum_{i=1}^d x_i \text{trace}(A_k B_i) = b_k \ (k = 1, \ldots, m), \sum_{i=1}^d x_i B_i \succeq 0 \right\}.$$

- Replace the LMI by $\sum_{i=1}^d x_i Q^* B_i Q \succeq 0$ to get block-diagonal structure.
- Delete any identical copies of blocks in the block structure.
A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra is called a coherent configuration if:

- The $B_i$’s are 0-1 matrices;
A basis $B_1, \ldots, B_d$ of a matrix $*$-algebra is called a **coherent configuration** if:

- The $B_i$’s are 0-1 matrices;
- For each $i$, $B_i^T = B_{i^*}$ for some $i^* \in \{1, \ldots, d\}$;
A basis $B_1, \ldots, B_d$ of a matrix $*$-algebra is called a coherent configuration if:

- The $B_i$'s are 0-1 matrices;
- For each $i$, $B_i^T = B_{i^*}$ for some $i^* \in \{1, \ldots, d\};$
- $\sum_{i=1}^d B_i = J$ (the all-ones matrix);
A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra is called a coherent configuration if:

- The $B_i$’s are 0-1 matrices;
- For each $i$, $B_i^T = B_{i^*}$ for some $i^* \in \{1, \ldots, d\}$;
- $\sum_{i=1}^d B_i = J$ (the all-ones matrix);
- $I \in \text{span}\{B_1, \ldots, B_d\}$.
Coherent configurations

A basis $B_1, \ldots, B_d$ of a matrix *-algebra is called a coherent configuration if:

- The $B_i$’s are 0-1 matrices;
- For each $i$, $B_i^T = B_{i^*}$ for some $i^* \in \{1, \ldots, d\}$;
- $\sum_{i=1}^d B_i = J$ (the all-ones matrix);
- $I \in \text{span}\{B_1, \ldots, B_d\}$.

If the $B_i$’s also commute, and $B_1 = I$, then we speak of an association scheme.
Coherent configurations

A basis $B_1, \ldots, B_d$ of a matrix $\ast$-algebra is called a coherent configuration if:

- The $B_i$’s are 0-1 matrices;
- For each $i$, $B_i^T = B_{i^*}$ for some $i^* \in \{1, \ldots, d\}$;
- $\sum_{i=1}^d B_i = J$ (the all-ones matrix);
- $I \in \text{span}\{B_1, \ldots, B_d\}$.

If the $B_i$’s also commute, and $B_1 = I$, then we speak of an association scheme.

Consequence

If $\mathcal{A}_{SDP}$ is spanned by a coherent configuration and $X = \sum_{i=1}^d x_i B_i$, then

$$X \succeq 0 \text{ and } X \succeq 0 \iff \sum_{i=1}^d x_i B_i \succeq 0 \text{ and } x \succeq 0.$$
Example

The **circulant matrices** have the basis

\[
\begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & \cdots & \vdots \\
0 & 0 & 1 & 0 & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & 1
\end{bmatrix},
\begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & \vdots \\
0 & 0 & 1 & \vdots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \cdots & 0 & 0 & 0
\end{bmatrix},
\ldots
\]

and form an association scheme.
How does one find a coherent algebra $\mathcal{A}_{SDP}$ that contains $\{A_0, \ldots, A_m\}$?
How does one find a coherent algebra $\mathcal{A}_{SDP}$ that contains $\{A_0, \ldots, A_m\}$?

An $O(n^3 \log n)$ algorithm is given in:

How does one find a coherent algebra $\mathcal{A}_{SDP}$ that contains $\{A_0, \ldots, A_m\}$?

An $O(n^3 \log n)$ algorithm is given in:


A more recent implementation (C code) is described in:

How does one find a coherent algebra $\mathcal{A}_{SDP}$ that contains $\{A_0, \ldots, A_m\}$?

An $O(n^3 \log n)$ algorithm is given in:


A more recent implementation (C code) is described in:


We give an illustration of how the method works ...
Example: Weisfeiler-Leman Algorithm

What is the smallest coherent algebra that contains this matrix?

$$
\begin{pmatrix}
0 & 3 & 2 & 2 & 4 & 4 \\
3 & 0 & 4 & 4 & 2 & 2 \\
2 & 4 & 1 & 4 & 4 & 4 \\
2 & 4 & 4 & 1 & 4 & 4 \\
4 & 2 & 4 & 4 & 1 & 4 \\
4 & 2 & 4 & 4 & 4 & 1
\end{pmatrix}
$$
Example: Weisfeiler-Leman Algorithm

What is the smallest coherent algebra that contains this matrix?

\[
\begin{pmatrix}
0 & 3 & 2 & 2 & 4 & 4 \\
3 & 0 & 4 & 4 & 2 & 2 \\
2 & 4 & 1 & 4 & 4 & 4 \\
2 & 4 & 4 & 1 & 4 & 4 \\
4 & 2 & 4 & 4 & 1 & 4 \\
4 & 2 & 4 & 4 & 4 & 1 \\
\end{pmatrix}
\]

Replace equal entries by different, non-commuting variables:

\[
\begin{pmatrix}
0 & 3 & 2 & 2 & 4 & 4 \\
3 & 0 & 4 & 4 & 2 & 2 \\
2 & 4 & 1 & 4 & 4 & 4 \\
2 & 4 & 4 & 1 & 4 & 4 \\
4 & 2 & 4 & 4 & 1 & 4 \\
4 & 2 & 4 & 4 & 4 & 1 \\
\end{pmatrix} \rightarrow \begin{pmatrix}
t_0 & t_3 & t_2 & t_2 & t_4 & t_4 \\
t_3 & t_0 & t_4 & t_4 & t_2 & t_2 \\
t_2 & t_4 & t_1 & t_4 & t_4 & t_4 \\
t_2 & t_4 & t_4 & t_1 & t_4 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_1 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_4 & t_1 \\
\end{pmatrix}.
\]
Example: Weisfeiler-Leman Algorithm (ctd.)

Now square the symbolic matrix to get:

\[
\begin{pmatrix}
t_0 & t_3 & t_2 & t_2 & t_4 & t_4 \\
t_3 & t_0 & t_4 & t_4 & t_2 & t_2 \\
t_2 & t_4 & t_1 & t_4 & t_4 & t_4 \\
t_2 & t_4 & t_4 & t_1 & t_4 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_1 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_4 & t_1
\end{pmatrix}
\]

\[=\]

\[
\begin{pmatrix}
x_0 & x_2 & x_3 & x_3 & x_4 & x_4 \\
x_2 & x_0 & x_4 & x_4 & x_3 & x_3 \\
x_5 & x_6 & x_1 & x_7 & x_8 & x_8 \\
x_5 & x_6 & x_7 & x_1 & x_8 & x_8 \\
x_6 & x_5 & x_8 & x_8 & x_1 & x_7 \\
x_6 & x_5 & x_8 & x_8 & x_7 & x_1
\end{pmatrix},
\]

where

\[
x_0 = t_0^2 + 2t_2^2 + t_3^2 + 2t_4^2
\]

\[
x_1 = t_1^2 + t_2^2 + 4t_3^2
\]

\[
x_2 = t_0 t_3 + 2t_2 t_4 + t_3 t_0 + 2t_4 t_2, \text{ etc.}
\]
Example: Weisfeiler-Leman Algorithm (ctd.)

Now square the symbolic matrix to get:

\[
\begin{pmatrix}
t_0 & t_3 & t_2 & t_2 & t_4 & t_4 \\
t_3 & t_0 & t_4 & t_4 & t_2 & t_2 \\
t_2 & t_4 & t_1 & t_4 & t_4 & t_4 \\
t_2 & t_4 & t_4 & t_1 & t_4 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_1 & t_4 \\
t_4 & t_2 & t_4 & t_4 & t_4 & t_1 \\
\end{pmatrix}^2 = \begin{pmatrix}
x_0 & x_2 & x_3 & x_3 & x_4 & x_4 \\
x_2 & x_0 & x_4 & x_4 & x_3 & x_3 \\
x_5 & x_6 & x_1 & x_7 & x_8 & x_8 \\
x_5 & x_6 & x_7 & x_1 & x_8 & x_8 \\
x_6 & x_5 & x_8 & x_8 & x_1 & x_7 \\
x_6 & x_5 & x_8 & x_8 & x_7 & x_1 \\
\end{pmatrix},
\]

where

\[
\begin{align*}
x_0 &= t_0^2 + 2t_2^2 + t_3^2 + 2t_4^2 \\
x_1 &= t_2^2 + t_3^2 + 4t_3^2 \\
x_2 &= t_0t_3 + 2t_2t_4 + t_3t_0 + 2t_4t_2, \text{ etc.}
\end{align*}
\]

Repeat this process and add new symbols until convergence.
And, finally ...

Symmetry reduction in SDP is the application of representation theory to reduce the size of specially structured SDP instances.
And, finally ...

- Symmetry reduction in SDP is the application of representation theory to reduce the size of specially structured SDP instances.
- The most notable applications are in computer assisted proofs (bounds on crossing numbers, kissing numbers, error correcting codes, ...)
And, finally ...

- Symmetry reduction in SDP is the application of representation theory to reduce the size of specially structured SDP instances.
- The most notable applications are in computer assisted proofs (bounds on crossing numbers, kissing numbers, error correcting codes, ...)
- ... but also pre-processing of some SDP’s arising in optimal design (truss design, QAP, ...
Symmetry reduction in SDP is the application of representation theory to reduce the size of specially structured SDP instances.

The most notable applications are in computer assisted proofs (bounds on crossing numbers, kissing numbers, error correcting codes, ...)

... but also pre-processing of some SDP’s arising in optimal design (truss design, QAP, ...)

More applications in polynomial optimization, graph coloring, ...
**Survey paper:**


**Further reading:**


THANK YOU!